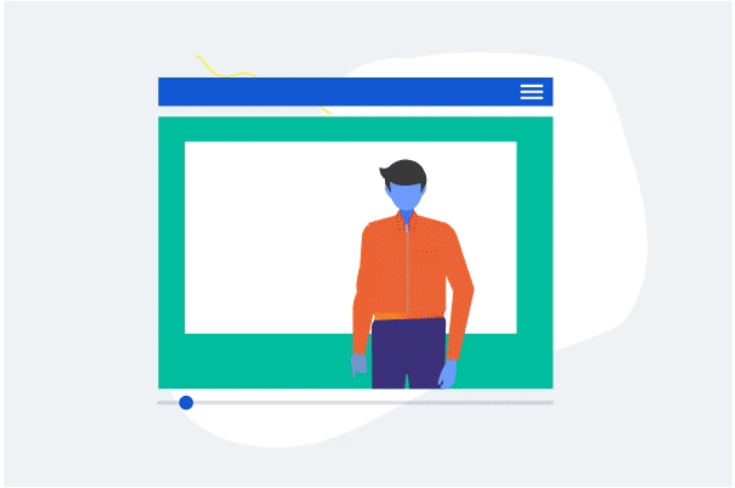

Day1 Labs

vlsideepdive

OpenSTA: Introduction

- OpenSTA is a gate level static timing verifier. As a stand-alone executable it can be used to verify the timing of a design using standard file formats.
  - Verilog netlist
  - Liberty library
  - SDC timing constraints
  - SDF delay annotation
  - SPEF parasitics
- OpenSTA is architected to be easily bolted on to other tools as a timing engine. By using a network adapter, OpenSTA can access the host netlist data structures without duplicating them.
- Query based incremental update of delays, arrival and required times
- Simulator to propagate constants from constraints and netlist tie high/low
- OpenSTA pdf doc
  - https://github.com/The-OpenROAD-Project/OpenSTA/blob/master/doc/OpenSTA.pdf

vlsideepdive

Basics of OpenSTA

- OpenSTA is a Static Timing analysis (STA) tool
- An STA tool takes design, standard cell, constraints as input and perform timing checks on the design
- The kind and type of checks we have covered in lectures
- Delay calculation
  - Integrated Dartu/Menezes/Pileggi RC effective capacitance algorithm
- External delay calculator API
- Analysis
  - Report timing checks -from, -through, -to, multiple paths to endpoint
  - Report delay calculation
  - Check timing setup

Inputs to OpenSTA

- Design
  - Netlist format
  - Usually provided in Verilog using read_verilog command
- Steps to do
  - cd to lab1 directory using following unix command “cd lab1”
  - Type “ls” and you will notice one of the file named “simple.v”
  - simple.v is our design in Verilog format
  - Open file using “leafpad simple.v”.
  - You would see the following, these are standard cells or lib cells instantiations

```
// Start cells
sky130_fd_sc_hd__nand2_1 u1 ( .A(inp1), .B(inp2), .Y(n1) );
sky130_fd_sc_hd__dfbbn_2 f1 ( .D(n2), .CLK_N(tau2015_clk), .Q(n3),
RESET_B(1'b1));
sky130_fd_sc_hd__inv_1 u2 ( .A(n3), .Y(n4) );
sky130_fd_sc_hd__inv_1 u3 ( .A(n4), .Y(out) );
sky130_fd_sc_hd__nor2_1 u4 ( .A(n1), .B(n3), .Y(n2) );
```

Standard
Cell Instantiations

Inputs to OpenSTA

- Standard Cells
  - Provided in .lib format using read_liberty command
  - A typical cell will have
    - Input ports definitions
    - Output port definitions
    - Functionality of the cells
    - Input to Output relationship
- Type “ls ..” and you will notice one of the file named “sky130_fd_sc_hd_tt_025C_1v80.lib”
  - This is our library cell information in .lib (liberty format)
  - Open file using “leafpad sky130_fd_sc_hd_tt_025C_1v80.lib”.
  - We will understand details of this format in later labs
  - Look for “sky130_fd_sc_hd__nand2_1” in this file
    - Notice this is the cell which was instantiated in our Verilog design
    - Find out what are the input and output pins of this cell

Inputs to OpenSTA

- Timing constraints
  - Provided in sdc format using read_sdc command

Constraints creation – Defining clocks

- Clocks are defined using create_clock command
  - Lets define a clock with period 50 on port tau2015_clk

```
create_clock -period 50 -name tau2015_clk [get_ports tau2015_clk]
```

Constraints creation

- IO delays-

  Primary ports are defined with delays with associated clock: `tau2015_clk`

```
set_input_delay 5 –max –rise [get_ports inp1] –clock tau2015_clk
set_output_delay -10 –min –fall [get_ports out] –clock tau2015_clk
```

vlsideepdive

Constraints creation

- Input transitions by environmental factors-

```
set_input_transition 10 –min –rise [get_ports inp1]
```

- Capacitive load on output pin-

```
set_load –pin_load 4 [get_ports out]
```

Constraints creation

- Type “ls” and you will notice one of the file named “simple.sdc”
- simple.sdc is our constraint file
- Open file using “leafpad simple.sdc”.

runScript

- In runscript you can define all the commands you want to run in the openSTA tool
- Tool will execute each command sequentially in order
  - There are some commands which are executed in parallel in some cases
- Runscript is in tcl format
- Type “ls” and you will notice one of the file named “run.tcl”
  - Open file using “leafpad run.tcl”.

```
## reading liberty model
read_liberty ../sky130_fd_sc_hd__tt_025C_1v80.lib
## reading netlist model
read_verilog simple.v
link_design simple
## Parsing constraints
read_sdc -echo simple.sdc
## report timing reports for 5 paths
report_checks -group_count 5
```

vlsideepdive

Run OpenSTA

- Run openSTA using command “sta run.tcl -exit | tee run.log”

- Fun things to do
  - Remove some commands from runscript see what you notice in the log file
  - Remove some commands from sdc and see what you notice in the log file